

Examining the last writer slices from failing and non-failing executions reveals the ordering violation.

Thread 1

```
void hts_mutexinit(htsmutex* mutex) {  
    htsmutex_s* smutex = malloc(sizeof(htsmutex_s));  
    pthread_mutex_init(&smutex, 0);  
    A *mutex = smutex;  
}
```

Thread 2

```
int hts_cancel_file_push(...) {  
    int ret;  
    hts_mutexlock(&opt->state.lock); B  
    ret = hts_cancel_file_push_(opt, url);  
    hts_mutexrelease(&opt->state.lock);  
    return ret; }
```

Information reported back to programmer

Non-Failing Run Memory Image @ B

&opt->state.lock = <valid mutex state>

Non-Failing Run Last Writer Slice @ B

LWS(&opt->state.lock) = Thread 1@ A

Failing Run Core dump @ B

&opt->state.lock = <garbage data> or NULL

Failing Run Last Writer Slice @ B

LWS(&opt->state.lock) = <uninitialized>